Chapter 3: Transport Layer

Chapter goals:
- understand principles behind transport layer services:
  - multiplexing/demultiplexing
  - reliable data transfer
  - flow control
  - congestion control
- instantiation and implementation in the Internet

Chapter Overview:
- transport layer services
- multiplexing/demultiplexing
- connectionless transport: UDP
  - principles of reliable data transfer
- connection-oriented transport: TCP
  - reliable transfer
  - flow control
  - connection management
  - principles of congestion control
  - TCP congestion control

Transport services and protocols
- provide logical communication between app' processes running on different hosts
- transport protocols run in end systems
- transport vs network layer services:
  - network layer: data transfer between end systems
  - transport layer: data transfer between processes (applications)
    - relies on, enhances, network layer services

Transport-layer protocols
- reliable, in-order unicast delivery (TCP)
  - congestion
  - flow control
  - connection setup
- unreliable ("best-effort"), unordered unicast or multicast delivery: UDP
- services not available:
  - real-time
  - bandwidth guarantees
  - reliable multicast

Multiplexing/demultiplexing
Recall: segment - unit of data exchanged between transport layer entities
- aka TPDU: transport protocol data unit
- aka 4-PDU

Demultiplexing: delivering received segments to correct app layer processes
Multiplexing/demultiplexing

- Gathering data from multiple app processes, enveloping data with header (later used for demultiplexing)

- Multiplexing/demultiplexing:
  - Based on sender, receiver port numbers, IP addresses
    - Source, dest port #s in each segment
    - Recall: well-known port numbers for specific applications

TCP/UDP segment format

- Source port #
- Dest port #
- Other header fields
- Application data (message)

Why is there a UDP?

- "No frills," "bare bones" Internet transport protocol
- "Best effort" service, UDP segments may be:
  - Lost
  - Delivered out of order to app
- Connectionless:
  - No handshaking between UDP sender, receiver
  - Each UDP segment handled independently of others

UDP: User Datagram Protocol [RFC 768]

- “No frills,” “bare bones” Internet transport protocol
- “Best effort” service, UDP segments may be:
  - Lost
  - Delivered out of order to app
- Connectionless:
  - No handshaking between UDP sender, receiver
  - Each UDP segment handled independently of others

UDP: more

- Often used for streaming multimedia apps
  - Loss tolerant
  - Rate sensitive
- Other UDP uses (why?):
  - DNS
  - SNMP
- Reliable transfer over UDP: add reliability at application layer
  - Application-specific error recovery!
UDP checksum

**Goal:** detect "errors" (e.g., flipped bits) in transmitted segment

**Sender:**
- treat segment contents as sequence of 16-bit integers
- checksum: addition (1's complement sum) of segment contents
- sender puts checksum value into UDP checksum field

**Receiver:**
- compute checksum of received segment
- check if computed checksum equals checksum field value:
  - NO - error detected
  - YES - no error detected.
    *But maybe errors nonetheless? More later....*